Learning to Assemble: Estimating 6D Poses for Robotic Object-Object Manipulation

Stefan Stevšić[1], Sammy Christen[1], Otmar Hilliges[1]

***Abstract*— In this paper we propose a robotic vision task with the goal of enabling robots to execute complex assembly tasks in unstructured environments using a camera as the primary sensing device. We formulate the task as an instance of 6D pose estimation of template geometries, to which manipulation objects should be connected. In contrast to the standard 6D pose estimation task, this requires reasoning about local geometry that is surrounded by arbitrary context, such as a power outlet embedded into a wall. We propose a deep learning based approach to solve this task alongside a novel dataset that will enable future work in this direction and can serve as a benchmark. We experimentally show that state-of-the-art 6D pose estimation methods alone are not sufficient to solve the task but that our training procedure significantly improves the performance of deep learning techniques in this context.**

I. Introduction

When manipulating complex objects, humans reason about shape *and* function. For example, humans know that screws fit into threaded nuts and know which type of tool should be used to tighten the screw based on its head. More generally, we are adept in understanding object interactions by observing the shape of important key parts of an object. In particular, humans can manipulate the same class of object irrespective of its context. We know how to use a power outlet, whether it is embedded into a plain wall or a complex piece of furniture. To enable robots to perform similar tasks, we can use a camera to estimate the 6D pose of the target into which further assembly components, such as tools or parts of compound objects, should be placed. The resulting predictions can then be used for planning of robot end-effector trajectories. However, the above type of generalization capability remains elusive for most methods.

Predicting 6D pose of assembly targets in varying contexts is hard because the texture and the surrounding shape can be drastically different from one instance to another. The different appearance of targets makes the detection of key features harder. Deep learning methods can learn better representations using context information. However, pose estimation can be biased by neighboring pixels, especially if the number of examples in the training dataset is small. We propose to overcome this issue via bootstrapping shape information from a single assembly example that is presented to the learner with randomized surrounding geometry and texture (see Fig. 1).

[1]AIT Lab, Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland stefan.stevsic | sammy.christen | otmar.hilliges @inf.ethz.ch

We thank the NVIDIA Corporation for the donation of GPUs used in this work.

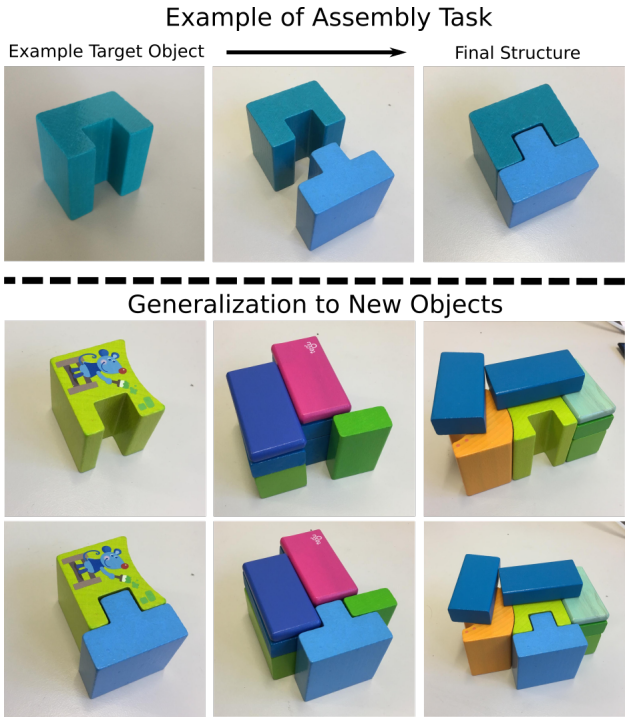

Fig. 1. Top: Example of an assembly task. Given a target object, the manipulation block should be placed to obtain a compound structure. Bottom: Given the example on the top, generalization should be possible as shown in the two bottom rows.

In this paper, we propose and study the new robotic vision task of 6D pose estimation for assembly scenarios. The goal is to predict the desired final poses for manipulation objects, using only raw RGB-D images as input. Approaches to learning-based 6D pose estimation of rigid objects [1], [2], [3] have recently made rapid progress in terms of single object accuracy. However, these approaches can only predict the pose of a fixed set of a priori known objects. In contrast, our goal is to attain a method that can generalize to objects that *contain* a shape template but are otherwise unseen. The key insight is that many assembly pieces always fit to a template geometry, which repeats across different objects (see Fig. 1). By letting the learner focus on the template geometry, it should be possible to achieve generalization to objects outside of the training set. Fig. 1, top shows an example of a typical assembly task studied here. Crucially, we introduce a task where a method is trained only on a single template geometry and needs to be able to generalize to new objects that *contain* the template (see Fig. 1, bottom).

To address issues in our task, we introduce a novel training scheme that leads to better generalization properties compared to standard 6D pose estimation methods. Our training data is generated synthetically, allowing for easy manipulation of geometry and texture. We show that by using only synthetic training data, where the texture of the object is randomized, the algorithm can predict the 6D pose with relatively high accuracy when tested on the same geometry with unknown texture. However, when presented with a new object that contains the template geometry, performance significantly drops. We address this issue by introducing a novel domain randomization technique, which randomizes the shape *surrounding* the target shape. As a proof-of-concept, we implement a simple open-loop control scheme on a real robot, which shows that our approach leads to the successful execution of assembly tasks.

To foster further progress on this task, we introduce and make publicly available a new dataset with synthetic images for training and real images for testing. To this end, we captured 24 different objects categorized into 4 different assembly tasks. The objects are composed of colored wooden blocks, yielding variability of shape and texture. We provide annotation of the 6D poses, the geometries of the manipulator objects and the target shapes, as well as RGB-D images. Since the utilized wooden blocks are readily available in any toy store, our setup can be easily rebuilt for testing on a robot.

In summary, we contribute the following: (i) We introduce and analyze a new task in the area of robotic vision. Solving the task might have important implications in assembly tasks. (ii) We propose a domain randomization technique that improves the accuracy of our task compared to the standard 6D pose estimation pipeline. We demonstrate that our approach leads to the completion of assembly tasks on a real-world robot. (iii) We introduce a training and a test dataset to evaluate our approach and enable other researchers to work on this task.

II. Related Work

In recent years, 6D pose estimation has received a lot of attention in the robotics and computer vision communities [1], [2], [3], [4], [5]. The long-term goal is to enable robots to manipulate objects in unstructured environments. Thus, estimating positions and orientations of manipulation objects is an important piece of the manipulation pipeline. Using hand crafted features [5], [6], it is possible to locate objects with a relatively high accuracy when objects are not occluded. With the emergence of Convolutional Neural Networks (CNNs), researchers have tried to improve the performance by training models to estimate the 6D pose directly from image data [1], [3], [4], [7]. However, these methods rely on a predefined set of objects, and thus cannot be applied directly to our task.

Several papers try to solve 6D pose estimation by training only on synthetic data [2], [8] by applying domain randomization approaches. In [8], the authors emphasize the importance of parameter randomization when synthetic data is generated and randomize parameters such as position and texture of objects, position and rotation of lights, background texture, and camera transformations. More recently, [2] shows that a high level of precision is possible by training only on synthetic data. Compared to previous work, [2] uses a 6D pose estimation specific neural network architecture and a significant amount of photo realistic scenes. Our work builds on their publicly available data generation pipeline NDDS, but extends the domain randomization scheme to geometry. Similar domain randomization techniques are used to bridge the domain gap in object detection tasks [9]. Furthermore, [10] show that by using domain randomization, transfer is possible to a real robot for pick and place tasks.

Research on assembly tasks mostly focuses on planning a sequence of manipulation actions with pre-defined objects [11]. The planner often relies on a set of rules that describe how parts can be connected together [12]. These systems work in isolation, relying on the availability of pose information of known parts. For example, the task of packing objects into boxes can be solved by using a camera as a sensing device, but only if the object and box shape are known a priori [13]. The authors of [14] show that it is possible to construct a tool by reasoning about the shape of the desired tool and the shape of its parts. Their focus is on assembly planning, and thus they only use rectangular wooden blocks which are easy to recognize in a structured environment. This work shows that geometric reasoning may be useful for assembly tasks in unstructured environments.

III. Problem Definition

In this paper we introduce the task of 6D template pose estimation for robotic assembly. To illustrate the problem, consider the task of placing a screw into a hole. To complete this task, the vision algorithm needs to detect the screw hole, from which the target 6D pose can be estimated. Once the goal pose is available, a planner can compute a trajectory for a robot arm to insert the screw. Thus, the crucial problem is to estimate the 6D target pose from the visual scene. 6D pose estimation is usually done in two steps: 1) the initial pose is estimated from the full image of the scene; 2) the pose is refined locally, using data from the image cropped around the template. We focus on the initial 6D pose estimate, since it influences the quality of the following steps and thus the final task performance. An illustration of the typical steps involved in assembly tasks is shown in Fig. 2.

The screw insertion task can be part of a more complex assembly task, where the target is embedded into different geometries. An example of such generalization is shown in Fig. 1. The T shaped block can be connected to different objects via the same template geometry. Note how the shape and texture of the target objects change in this example. To achieve generalization, we can exploit the existence of repeating geometric patterns of template geometries.

More precisely, we pose the problem as a prediction of the template 6D pose $\widetilde{\mathbf{T}} = [\widetilde{\mathbf{R}}|\widetilde{\mathbf{t}}]$ directly from pixel data. Since our goal is to enable generalization to different scenes, we use the following experimental setup: at training time, only a pair of assembled objects (c.f. Fig 1, top) are available.

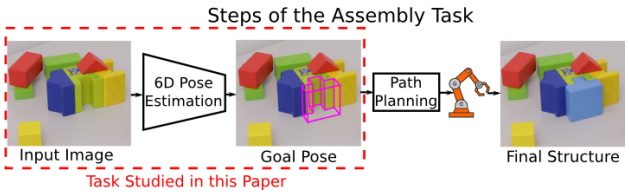

Fig. 2. The steps of an assembly task. First, the goal pose is predicted from a raw image, which is the focus of this paper. The goal pose is then used to plan the path of the robot arm.

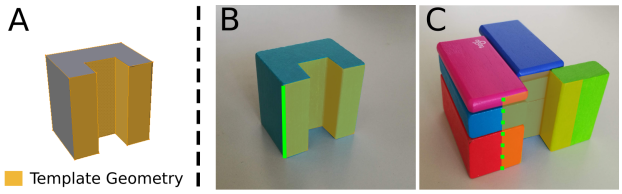

Fig. 3. **A**: 3D model of the example target object with the template geometry. This mesh is used to generate the training data. **B**: Real target object containing the template geometry in a similar context. Green line shows a visible template edge. **C**: Real target object with the template geometry in a different context. The edge from B is not visible (green dashed line).

One of these objects is the *manipulation object* (e.g., the T-shaped object on Fig 1, top), and the other is the *example target object*. We define the *template geometry* as a part of the target object mesh (see Fig. 3, B). The manipulation object fits exactly into the template geometry. Therefore the template can be represented by a contact profile (see shaded area in Fig. 3), irrespective of the surrounding geometry. At test time, the scene can contain target objects with a template that is surrounded by different shapes and textures compared to the example target object (see Fig. 3, C).

Estimating the 6D pose of the template is a difficult task. The templates are small and most of the time self-occluded. Moreover, the template is embedded into a surrounding geometry with unseen texture, which results in different appearances in images. Generally speaking, 6D pose estimation is posed as either regression [1], [3] or classification [15], [5] of poses from image data. The appearance of the template changes substantially with the rotation of the object, which makes the problem hard even if the template is fully visible. Occlusions and the influence of surrounding geometry make the problem even harder. Surrounding pixels drastically change the appearance of the template, making the detection of crucial features difficult. For instance, on the example target object (see Fig. 3. B) the left edge can be extracted via an edge detector. However, on the test example (see Fig. 3. C) this edge is not visible. Another difficulty is that we cannot rely on the texture, and thus the 6D pose needs to be inferred from the broader context in the image.

The advent of CNNs brought significant improvements on the 6D object pose estimation task [1], [2], [3], [4], [7]. However, generalizing over different textures and removing influence of the context are challenging problems for CNN architectures. CNNs are known to exhibit biases towards texture [16], especially if pretrained on ImageNet, which is the de-facto standard in 6D pose estimation methods [1], [2], [3]. Furthermore, deep CNN architectures possess a large receptive field and thus neighboring pixels of the template geometry have a strong influence on the pose estimation outcome. In this paper we provide empirical evidence (see Sec VI-B) that by naively applying 6D pose estimation approaches, such models will overfit to the specific object.

IV. Approach and Datasets

We argue that it is key to reason about the shape of the template geometry itself, minimizing the influence of the texture and the surrounding geometry. Thus, we propose a number of measures to overcome the inherent biases in existing methods for 6D pose estimation. Crucially, we propose a training scheme that can be applied to different existing methods and improves generalization to unseen objects (see Sec. VI-B). To evaluate our approach, we collect a dataset with real objects used in four different assembly scenarios.

Our approach relies entirely on synthetically generated data. However, we show that the approach does not lead to an insurmountable domain gap, via a test set containing only real images. The key ingredients of our approach are: 1) texture randomization and 2) shape randomization at train time. To test our approach, we re-implement the network architecture from [3], but our approach applies to other methods too (see Sec. VI-B). The method from [3] predicts the segmentation mask of a region of interest, the template geometry in our case. By selecting only the region of relevant features, the network is trained to reject information outside of this region. While it has previously been shown that such region of interest masks improve 6D pose estimation accuracy [3], [17], it is not sufficient to solve the problem studied here. This is due to the influence of data biases and we show experimentally that training a segmentation based method on data that contains only example objects does not generalize well (c.f Sec. VI-B).

A. Domain Randomization

To overcome the issue of texture biases, we randomize the appearance of the target object via application of different colors, material roughness, grid patterns, and random textures from [18]. We show experimentally that texture randomization results in high accuracy when tested on an object of the same shape. The texture of the test object is unknown, which confirms that the CNN can learn to estimate the pose of a particular shape by applying texture randomization. Similar findings were previously reported [16] to introduce a shape bias in models trained on the ImageNet dataset.

However, it is important to note, that despite texture randomization, performance on target objects of different shapes remains poor (see Sec. VI-B). Since the training data only contains a single example object, a strong shape bias is introduced. Standard domain randomization techniques cannot remove this bias. Increasing clutter around the target object in training data improves performance when the target is surrounded by other objects [19]. In our case, the template

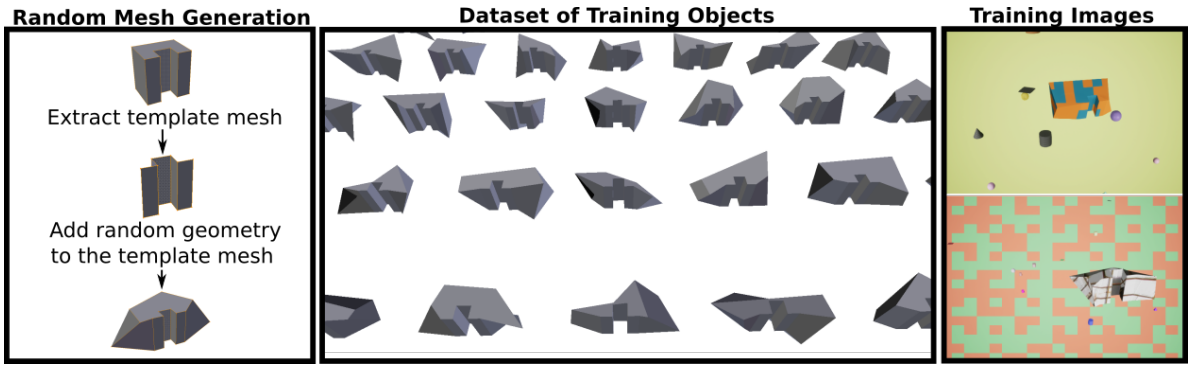

Fig. 4. Generation of the random mesh. The template mesh is extracted from the example object and a random surrounding mesh is added. Using this method, the dataset of training objects is constructed and then used to generate training images.

is embedded in the structure, and thus adding separate objects does not improve the result. To address this problem, we combine texture randomization with an additional step of shape randomization. The idea is to randomize the shape surrounding the template geometry. Thus, the network is forced to ignore the context and must focus on the template geometry which is held constant during training. To attain random shapes, we extract the template mesh (cf. Fig. 4, left). Next, we define a perturbed context via randomizing the positions of the vertices of a cuboid. Since every vertex position is changed independently, we obtain diverse context shapes (cf. Fig. 4, middle). The range of vertex positions is chosen such that the faces of the smallest possible cuboid do not intersect with the template mesh, while the upper range is selected from a much larger range of values. The random context shape is then connected to the template mesh analogously to the original context to produce watertight meshes. In total we generate 50 random objects per target shape. We experimentally determined that using more than 50 random objects yield no further improvement.

The NDDS pipeline [2] has been extended in order to produce the synthetic data. We used all domain randomization techniques, i.e., background randomization, light randomization, pose and rotation randomization, and randomly positioned distractor objects as described in [2]. We limit the rotation range to prevent full occlusion of the template. Furthermore, the rotation angles are restricted by the template angle of symmetry to prevent symmetric poses in the dataset, which has been show to improve accuracy [4]. For each object we generate 12000 training images with ground truth annotations, i.e., the position and the rotation of the template geometry, the positions of the 3D keypoints from a bounding box positioned around the template, 2D projections of the keypoints in the training image, and the template segmentation mask. In [9] it has been observed that more training samples did not improve the results significantly. We generate a few datasets to examine texture and shape randomization effects independently as described in Sec. VI.

B. Evaluation Dataset

For evaluation of our approach and as a baseline for future work, we propose four different exemplar assembly tasks, illustrated in Fig. 5. Every task corresponds to different manipulation objects with various types of geometries and with different types of (self-)occlusion. In Example 1, the goal is to place the cube into a corner, which displays a minimal L-shaped template geometry, as is typical in packing scenarios. Examples 2 and 3 contain more complex rectangular and cylindrical geometries. Example 4 corresponds to an inserting task where the template geometry is always at least partially occluded.

For each of the tasks, we recorded six different test scenes with different objects positioned on a desk. Each scene contains a different target object – either a single wooden block or a compound of different wooden blocks. The first object is always the example target object, which is available at training time (see Fig. 5). In 6D pose estimation, the object is available a priori. Thus, we use this example for comparison with standard 6D pose estimation settings, to evaluate the influence of surrounding shape and texture. The other five target objects have a different shape that is not available at training time. Examples of these objects are shown in Fig. 5, bottom row. To emulate natural settings, we place several distractor objects in the scene (see Fig. 8).

1) Dataset Characteristics: The test scenes have been recorded with an Intel RealSense RGB-D camera. The image resolution is 640x480. To measure the ground truth goal poses, we position the target at a predefined offset from a set of markers (Aruco) that are used to track the camera pose [20]. The use of markers results in precise tracking of the goal pose. We move the camera around the target objects with varying camera angles and distances to the target object. We keep the camera in a range corresponding to typical robotic manipulation tasks where a camera is attached to the robot. We rotate the camera between $\pm 90^\circ$ around the template. Moving the camera further results in almost complete occlusion of the template geometry, which would make detection very difficult. We provide the ground truth goal positions, RGB-D inputs, example objects' meshes, template meshes and manipulation objects' meshes as annotation. The test data contains 36000 images. We additionally provide our synthetic training data with all randomized geometries used to produce the data. In total there are 148000

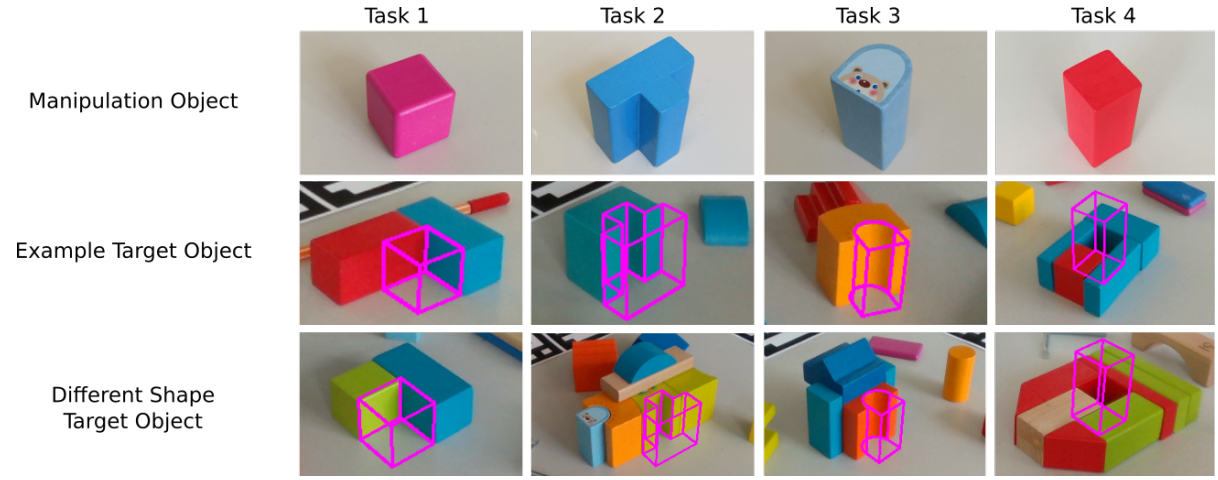

Fig. 5. The test dataset with ground truth annotations shown with pink wireframes. For each task we show the example object and one of the test objects with a different shape. Notice that in task 4, the wooden block should be placed inside the hole.

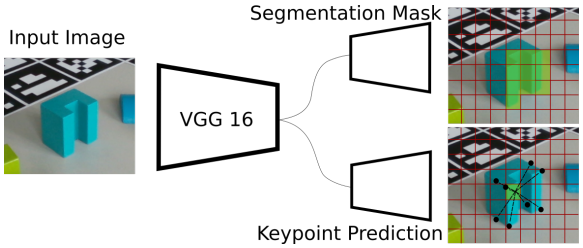

Fig. 6. Convolutional neural network used in our experiments. The segmentation mask is used to select 2D keypoint predictions only from within the template geometry.

images. The dataset is available on the project webpage: https://ait.ethz.ch/projects/2020/template6d/.

V. Implementation

In this section, we explain the specific implementation details. We discuss the structure of the neural network model, the refinement procedure, and the real robot implementation.

A. Network Details

In our experiments we follow [3] a CNN with two decoder streams: a segmentation mask, and a separate prediction stream for the bounding box keypoints with confidence estimation. Both streams use the same base network, in our case VGG 16, as shown in Fig. 6. We combine the outputs of the blocks 4 and 5 from VGG 16 and pass them to the decoders, similar to [1]. The output of the decoders are 3D tensors with spatial resolution $S \times S$ and feature dimensions D_{seg} and D_{reg}. The resolution $S \times S$ is lower than the original image resolution, in our case 50×50. This effectively means that we place an $S \times S$ grid onto the image (see Fig. 6). Each grid cell then produces D_{seg} segmentation votes, in our case only two, corresponding to background and template, and D_{reg} real numbers that predict the 2D keypoint locations and confidence scores. For k keypoints, D_{reg} is $3 \times k$, where $2 \times k$ are the predictions of x and y coordinates, and additionally k confidence predictions. Only predictions that fall within the segmentation mask are selected, both at training and test time. By masking out the predictions from the background, the network focuses on features from the template. This is especially important in the case of occlusions, where this regularization helps to remove the features from occluding objects which bias the prediction.

Each grid cell predicts $k = 8$ keypoints, corresponding to the 2D projections of the 3D bounding box corners. As suggested in [3], we use a RANSAC version of the PnP algorithm [21] to match the corners of the 3D bounding box with the 2D keypoints and to compute the template pose. The RANSAC algorithm finds the best match when multiple predictions are available. Thus, we select the 10 most confident predictions for each of the 8 keypoints. The algorithm can be extended to multiple targets via clustering of the keypoint predictions [3]. However, our test dataset contains only a single test object per scene, which enables us to isolate the influence of different contexts.

B. Refinement Step

To further improve the pose estimation accuracy, we use the Iterative Closest Point (ICP) algorithm. The ICP algorithm improves the pose by iteratively minimizing the distance between the 3D model of the template and point cloud obtained from a depth image. The accuracy of the ICP algorithm depends on the initial pose, which we obtain via the method described above. Therefore, improving the initial pose estimate also results in better final pose predictions.

C. Implementation on the Robot

For the robot experiments, the camera is placed on a fixed position facing the table upon which the objects are placed. The offset between the camera and robot is obtained in an off-line calibration procedure. To remove the influence of grasping errors, the manipulation objects are placed manually into the gripper of the ABB YuMi robot. The camera captures a single RGB-D frame, which is then used to compute the target pose. The robot arm trajectory is computed via the trajectory optimization algorithm from [22].

VI. Results

In this paper, we propose a new robotic vision task and several domain randomization techniques to improve generalization to unseen template context and appearance. We now evaluate our domain randomization techniques and point to the most critical issues of the proposed task. First, we show that via texture randomization, the network can learn to detect the 6D pose of a shape. Second, we show that the influence of the surrounding geometry is an important issue that cannot be solved by state-of-the-art CNNs for 6D pose estimation. We address this issue by our novel shape randomization technique and achieve significant improvement over the baselines. Finally, we show that by using our approach it is possible to execute the assembly task on the real robot.

To evaluate our approach, we generated two synthetic training datasets. To isolate the effects of texture randomization, the first training dataset uses only the texture randomization technique while keeping the geometry of the example target object. The second dataset uses the full domain randomization pipeline. For each task, we train a separate neural network. Inference time for both models is approximately 30 ms. Computing the pose via PnP algorithm is 4 ms and pose refinement takes 140 ms approximately. The full pipeline runs at 6 fps on a Nvidia Pascal Titan X.

For evaluation, we use our test dataset with real images as described in Sec. IV-B. We evaluate our approach separately on the example target object and compare results to the cases when the shape of the object is unknown a priori. This comparison highlights the influence of the surrounding shape on the pose prediction. Nevertheless, our primary goal is to achieve high accuracy on target objects with unseen shapes. Therefore, our main focus is on the evaluation of our approach on the objects 2-6 in each of the tasks.

A. Metrics

To evaluate the task performance, we propose two metrics: average distance (ADD) and ADD-S for symmetric objects. These metrics have been used routinely for 6D pose estimation [1], [2], [3], [5], [4], [7]. The ADD metric is defined as the mean distance between the 3D model points $\mathbf{x}$, transformed by the ground truth rotation $\mathbf{R}$ and translation $\mathbf{t}$, and the estimated rotation $\widetilde{\mathbf{R}}$ and translation $\widetilde{\mathbf{t}}$:

$$\text{ADD} = \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{M}} ||(\mathbf{R}\mathbf{x} + \mathbf{t}) - (\widetilde{\mathbf{R}}\mathbf{x} + \widetilde{\mathbf{t}})||. \quad (1)$$

In the previous equation, n is the number of points in the set $\mathcal{M}$ of 3D model points. In our case, we sample the points $\mathbf{x}$ uniformly from the 3D bounding box positioned around the template, which correspond to the space into which the manipulation object fits. The ADD-S metric is defined analogously:

$$\text{ADD-S} = \frac{1}{n} \sum_{\mathbf{x_1} \in \mathcal{M}} \min_{\mathbf{x_2} \in \mathcal{M}} ||(\mathbf{R}\mathbf{x_1} + \mathbf{t}) - (\widetilde{\mathbf{R}}\mathbf{x_2} + \widetilde{\mathbf{t}})||. \quad (2)$$

Compared to the ADD metric, ADD-S uses the distances between the closest points rather than specific pairs of points.

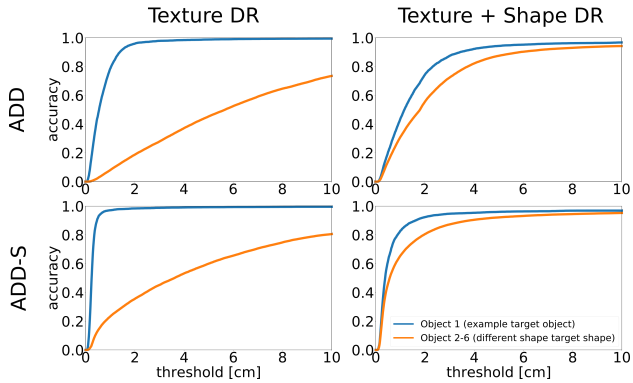

Fig. 7. Accuracy-threshold curves. The blue curves show average accuracy for the example objects, and the orange curves for objects with unseen shapes. The texture randomization accuracy significantly drops for objects with unseen shapes, while the full randomization results in high accuracy for all objects. We only show results for the method from [3] for clarity.

The final measure of performance is the ratio of correct pose predictions over the test set $\frac{N_c}{N_d}$, where N_c is number of correctly estimated poses and N_d is the number of samples in the dataset. We use two different thresholds to classify the pose as correct. The first threshold is used to evaluate if the pose is correctly initialized for the refinement procedure. This threshold is set to 2 cm. The second threshold indicates that the pose is correctly estimated. This threshold is 10% of the bounding box diameter, as commonly used in 6D pose estimation literature [3], [5], [4], [7]. We refer to this threshold as 0.1d. Keep in mind that objects are relatively small and that diameters are in the range of $5\ cm$ to $8\ cm$.

B. Evaluation

Additionally to our reimplementation of [3], we evaluate our domain randomization approach on a method from [2]. This model shows competitive performance on the task of 6D pose estimation when trained only on the synthetic dataset [2]. In the first two experiments, we evaluate the effects of texture and shape randomization on the neural network predictions without using the refinement step. In the last experiment, we evaluate the final precision of the algorithm after the ICP refinement step.

1) Texture randomization: When we use only texture randomization, the accuracy is satisfactory only for the test objects with known geometry (see Fig. 7). However, on other objects, texture randomization performs much worse. This experiment reveals two essential conclusions. First, these results suggest that the network indeed learns to recognize shapes via texture randomization. When the shape is fixed, the network predicts correct poses even though the textures are not available at training time. Second, the surrounding mesh strongly influences the 6D pose estimation of the template geometry. There is a significant gap between success rates on the objects with known shapes and the objects with unseen shapes. Even with the state-of-the-art method that rejects the influence of surrounding pixels via a segmentation mask [3], the network overfits to the training shape.

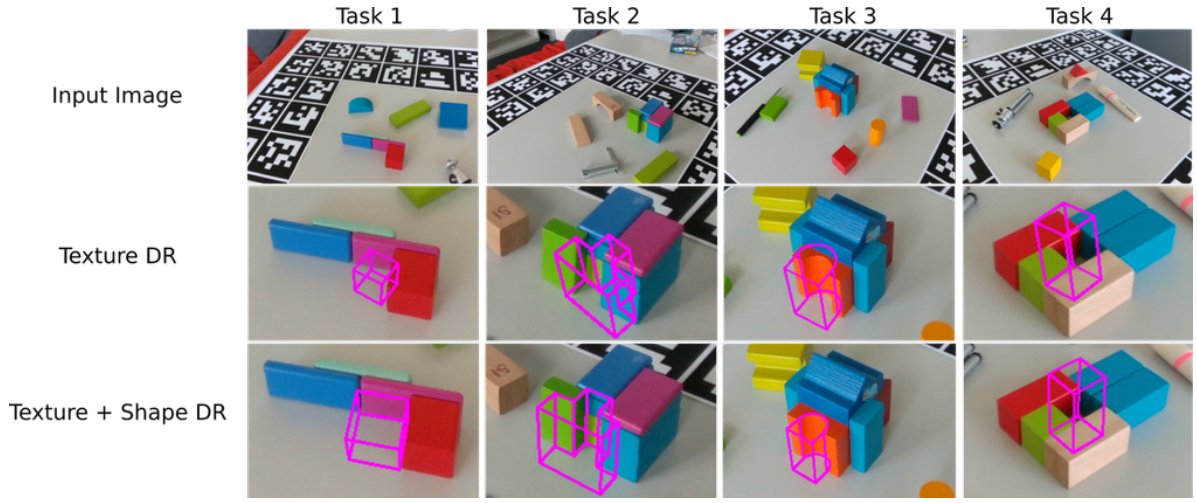

Fig. 8. Example predictions of our method, trained with different levels of domain randomization. Predictions are represented with pink wireframes. Note that predictions are rendered without z-culling and hence occlusions appear incorrectly in the right most case.

TABLE I

ADD SUCCESS RATES ON OBJECTS 2-6 (DIFFERENT SHAPE OBJECTS).

	Model from [2]				Model from [3]			
	Tex. DR		Tex.+Shape DR		Tex. DR		Tex.+Shape DR	
	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm
Task 1	3.0	11.1	**11.4**	**44.1**	0.5	7.3	**8.1**	**35.4**
Task 2	1.0	4.9	**22.3**	**59.3**	3.9	16.0	**20.4**	**58.3**
Task 3	14.2	30.8	**17.7**	**55.5**	10.5	42.5	**19.8**	**64.4**
Task 4	0.1	4.1	**16.2**	**68.1**	0.3	9.0	**19.5**	**63.5**
Mean	4.6	12.7	**16.9**	**56.8**	3.8	18.7	**17.0**	**55.4**

TABLE II

ADD-S SUCCESS RATES ON OBJECTS 2-6 (DIFFERENT SHAPE OBJECTS).

	Model from [2]				Model from [3]			
	Tex. DR		Tex.+Shape DR		Tex. DR		Tex.+Shape DR	
	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm
Task 1	8.8	21.5	**37.9**	**68.2**	4.9	21.3	**29.1**	**63.7**
Task 2	4.4	14.3	**60.1**	**86.9**	16.5	30.8	**58.6**	**82.8**
Task 3	27.5	39.9	**47.6**	**74.3**	35.1	60.9	**55.5**	**85.1**
Task 4	6.0	17.8	**66.7**	**95.3**	9.1	31.5	**64.5**	**92.0**
Mean	11.7	23.4	**53.1**	**81.2**	16.4	36.1	**51.9**	**80.9**

2) Full randomization: Our final goal is to have a method that works with objects of unseen shape. Therefore, the majority of our test objects have shapes that are not available at training time. The results are summarized in Table I and Table II. Note that in this experiment, the 2 cm threshold is more significant. This an important threshold beyond which a refinement procedure will no longer converge. A qualitative comparison can be seen in Fig. 8. To mitigate the influence of the surrounding shape, we add the shape randomization to the dataset generation pipeline. Shape randomization minimizes the influence of the surrounding shape by removing the surrounding shape biases in the training dataset. Our full domain randomization pipeline significantly improves the results irrespective of the neural network model used (see Table I and Table II). In case of the model from [3], the accuracy increases by 36.7 % and 45.0 % for the 2 cm threshold, for ADD and ADD-S errors respectively.

3) ICP refinement: As a refinement step, we implemented the ICP algorithm. The estimated goal pose is sent to the robot planner after the refinement step. Thus, we should

TABLE III

ADD SUCCESS RATES ON OBJECTS 2-6 (DIFFERENT SHAPE OBJECTS) AFTER ICP REFINEMENT.

	Model from [2] + ICP				Model from [3] + ICP			
	Tex. DR		Tex.+Shape DR		Tex. DR		Tex.+Shape DR	
	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm
Task 1	6.1	16.0	**23.4**	**62.2**	4.5	18.5	**25.6**	**61.7**
Task 2	8.7	15.9	**69.1**	**82.9**	23.0	32.5	**67.6**	**81.2**
Task 3	23.9	39.7	**36.5**	**70.8**	30.7	60.6	**45.6**	**83.2**
Task 4	1.1	14.2	**25.2**	**85.6**	2.3	22.6	**26.9**	**81.7**
Mean	9.9	21.5	**38.5**	**75.4**	15.1	33.5	**41.4**	**77.0**

TABLE IV

ADD-S SUCCESS RATES ON OBJECTS 2-6 (DIFFERENT SHAPE OBJECTS) AFTER ICP REFINEMENT.

	Model from [2] + ICP				Model from [3] + ICP			
	Tex. DR		Tex.+Shape DR		Tex. DR		Tex.+Shape DR	
	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm	<0.1d	<2 cm
Task 1	15.4	28.7	**61.6**	**73.9**	18.1	32.9	**60.8**	**73.2**
Task 2	16.5	20.9	**83.9**	**90.1**	33.8	40.6	**82.3**	**89.4**
Task 3	39.1	47.0	**69.8**	**82.2**	60.0	69.2	**82.3**	**92.5**
Task 4	14.6	28.0	**86.6**	**96.6**	25.2	41.9	**83.7**	**93.8**
Mean	21.4	31.2	**75.5**	**85.7**	34.3	46.2	**77.3**	**87.2**

pay attention to the 0.1d threshold, which indicates that the pose is correctly estimated. In Tables III and IV, we can notice a significant improvement when ICP is applied to the prediction obtained from the full domain randomization pipeline. After the refinement step, the pose estimate is more reliable compared to the initial estimation, implying that the output can be used for motion planning in assembly tasks.

When using the full randomization pipeline, both methods perform similarly in most of the tasks except in Task 3, where the method from [3] performs better. The half-cylinder 3D model is challenging for ICP refinement because the model can easily slide along the curvature. The method from [3] uses the RANSAC version of PnP which provides a more robust initialization for ICP. Another benefit of the method from [3] is that the keypoints are predicted directly from the network, while the method from [2] produces the keypoint heatmaps. It is a known issue that extracting a maximum from the down-sampled heatmap can be unstable.

Fig. 9. Example of the assembly task executed on the robot. The camera on the left captures the scene. From the image, our approach predicts the goal pose, which is used to compute the robot arm trajectory.

C. Experiments on a Real-World Robot

To demonstrate that our approach leads to the successful execution of assembly tasks, we implemented our algorithm on a real robot (c.f. Fig. 9). Please refer to the accompanying video for more results (url: https://youtu.be/hUyjFG7W5hM). Our system predicts a goal pose from a single image and produces a trajectory for the robotic gripper. The algorithm has the lowest error rates if the template is oriented towards the camera. In these cases, the robot can complete the task. However, the robot cannot always execute the task since the accuracy of 6D pose estimation decreases for more extreme viewing angles. By improving the method for 6D pose of template geometries or by adding a feedback loop, the accuracy could be improved.

VII. Discussion and Conclusion

In this paper, we introduce a robotic vision task that can have significant implications in assembly tasks. We analyze the most critical issues in this task, i.e., the influence of the unseen surrounding geometry to the 6D pose estimation. To address this problem, we introduce a domain randomization approach and show significant improvements irrespective of the underlying network architecture. Furthermore, we demonstrate that our pipeline leads to the successful execution of the assembly tasks on a real-world robot.

We present the initial work that enables generalization in assembly tasks via template geometries. To allow other researchers to improve the solution of the proposed task, we will release our training and test datasets. We suggest a few directions that could lead to better results. The accuracy of the 6D pose estimation can be improved in different ways, either by improving the training dataset, the neural network model, or improving the training loss function. Furthermore, by using closed-loop control, the planning algorithm can be improved to correct for the errors of the 6D pose estimation.

Acknowledgment

The authors would like to thank Simon Zimmermann and Miguel Angel Zamora Mora from CRL, ETH Zurich for their help with the implementation of the real robot demonstrator.

References

[1] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *arXiv preprint arXiv:1711.00199*, 2017.

[2] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[3] Y. Hu, J. Hugonot, P. Fua, and M. Salzmann, "Segmentation-driven 6d object pose estimation," *arXiv preprint arXiv:1812.02541*, 2018.

[4] M. Rad and V. Lepetit, "Bb8: a scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *ICCV*, 2017.

[5] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Asian conference on computer vision*. Springer, 2012, pp. 548–562.

[6] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *International conference on computer vision*. IEEE, 2011, pp. 858–865.

[7] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *International Conference on Computer Vision*, 2017, pp. 1521–1529.

[8] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *IROS*, 2017.

[9] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *CVPR Workshops*, 2018.

[10] S. James, A. J. Davison, and E. Johns, "Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task," *arXiv preprint arXiv:1707.02267*, 2017.

[11] W. Wan, K. Harada, and K. Nagata, "Assembly sequence planning for motion planning," *arXiv preprint arXiv:1609.03108*, 2016.

[12] I. Rodriguez, K. Nottensteiner, D. Leidner, M. Kabecker, F. Stulp, and A. Albu-Schaffer, "Iteratively refined feasibility checks in robotic assembly sequence planning," *RA-L*, vol. PP, pp. 1–1, 01 2019.

[13] R. Shome, W. N. Tang, C. Song, C. Mitash, H. Kourtev, J. Yu, A. Boularias, and K. E. Bekris, "Towards robust product packing with a minimalistic end-effector," *arXiv preprint arXiv:1903.00984*, 2019.

[14] L. Nair, J. Balloch, and S. Chernova, "Tool macgyvering: Tool construction using geometric reasoning," *arXiv preprint arXiv:1902.03666*, 2019.

[15] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3109–3118.

[16] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," *arXiv preprint arXiv:1811.12231*, 2018.

[17] M. Oberweger, M. Rad, and V. Lepetit, "Making deep heatmaps robust to partial occlusions for 3d object pose estimation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 119–134.

[18] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, "Describing textures in the wild," in *CVPR*, 2014.

[19] C. Mitash, K. E. Bekris, and A. Boularias, "A self-supervised learning system for object detection using physics simulation and multi-view pose estimation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 545–551.

[20] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[21] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, p. 155, 2009.

[22] S. Zimmermann, R. Poranne, J. M. Bern, and S. Coros, "Puppetmaster: robotic animation of marionettes," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, p. 103, 2019.